Importing the Dependencies

```
# loading
import pandas as pd
import numpy as np
# Data visualization
import matplotlib.pyplot as plt
import seaborn as sns
#preprocessing
from sklearn.preprocessing import StandardScaler, LabelEncoder
from collections import Counter
# Classification
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.ensemble import RandomForestClassifier
```

Data Collection

```
from google.colab import drive
drive.mount('/content/drive')

# Navigate to the file path and read it
import pandas as pd
file_path = '/content/drive/My Drive/csv file/city_day.csv'

# link of the dataset
# https://drive.google.com/file/d/1AaVlRYHos_jZzjYz22l0oTQYpaCfqSCf/view?usp=drive_link
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remou
data = pd.read_csv(file_path)
data.head()
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucke
0	Ahmedabad	2015-01-01	NaN	NaN	0.92	18.22	17.15	NaN	0.92	27.64	133.36	0.00	0.02	0.00	NaN	Na
1	Ahmedabad	2015-01-02	NaN	NaN	0.97	15.69	16.46	NaN	0.97	24.55	34.06	3.68	5.50	3.77	NaN	Na
2	Ahmedabad	2015-01-03	NaN	NaN	17.40	19.30	29.70	NaN	17.40	29.07	30.70	6.80	16.40	2.25	NaN	Na

Next steps: Generate code with data View recommended plots New interactive sheet

```
data.shape
```

```
(29531, 16)
```

Data Cleaning

```
data.columns
```

```
Index(['City', 'Date', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2',
       'O3', 'Benzene', 'Toluene', 'Xylene', 'AQI', 'AQI_Bucket'],
      dtype='object')
```

```
data.rename(columns = {'AQI_Bucket':'Air_quality'}, inplace = True)
```

```
cities = data['City'].value_counts()
print(f'Total number of cities in the dataset : {len(cities)}')
print(cities.index)
```

```
Total number of cities in the dataset : 26
Index(['Ahmedabad', 'Delhi', 'Mumbai', 'Bengaluru', 'Lucknow', 'Chennai',
       'Hyderabad', 'Patna', 'Gurugram', 'Visakhapatnam', 'Amritsar',
       'Jorapokhar', 'Jaipur', 'Thiruvananthapuram', 'Amaravati',
       'Brajrajnagar', 'Talcher', 'Kolkata', 'Guwahati', 'Coimbatore',
       'Shillong', 'Chandigarh', 'Bhopal', 'Ernakulam', 'Kochi', 'Aizawl'],
      dtype='object', name='City')
```

```
# Convert string to datetime64
data['Date'] = pd.to_datetime(data['Date'])
#data.set_index('Date',inplace=True)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29531 entries, 0 to 29530
Data columns (total 16 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   City         29531 non-null  object
 1   Date         29531 non-null  datetime64[ns]
 2   PM2.5        24933 non-null  float64
 3   PM10         18391 non-null  float64
 4   NO           25949 non-null  float64
 5   NO2          25946 non-null  float64
 6   NOx          25346 non-null  float64
 7   NH3          19203 non-null  float64
 8   CO           27472 non-null  float64
 9   SO2          25677 non-null  float64
 10  O3           25509 non-null  float64
 11  Benzene      23908 non-null  float64
 12  Toluene      21490 non-null  float64
 13  Xylene       11422 non-null  float64
 14  AQI          24850 non-null  float64
 15  Air_quality  24850 non-null  object
dtypes: datetime64[ns](1), float64(13), object(2)
memory usage: 3.6+ MB
```

```
# Check for missing values
print(data.isnull().sum())
```

```
City               0
Date               0
PM2.5           4598
PM10           11140
NO              3582
NO2             3585
NOx             4185
NH3            10328
CO              2059
SO2             3854
O3              4022
Benzene         5623
Toluene         8041
Xylene         18109
AQI             4681
Air_quality     4681
dtype: int64
```

```
# Fill missing values only for numeric columns
numeric_cols = data.select_dtypes(include=[np.number]).columns
data[numeric_cols] = data[numeric_cols].fillna(data[numeric_cols].median())
```

```
# Example of filling missing values in categorical columns with mode
categorical_cols = data.select_dtypes(include=['object']).columns
for col in categorical_cols:
    data[col].fillna(data[col].mode()[0], inplace=True)
```

```
data.head()
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	Air_qua
0	Ahmedabad	2015-01-01	48.57	95.68	0.92	18.22	17.15	15.85	0.92	27.64	133.36	0.00	0.02	0.00	118.0	Moc
1	Ahmedabad	2015-01-02	48.57	95.68	0.97	15.69	16.46	15.85	0.97	24.55	34.06	3.68	5.50	3.77	118.0	Moc
2	Ahmedabad	2015-01-03	48.57	95.68	17.40	19.30	29.70	15.85	17.40	29.07	30.70	6.80	16.40	2.25	118.0	Moc

Next steps: Generate code with data | View recommended plots | New interactive sheet

Exploratory Data Analysis (EDA)

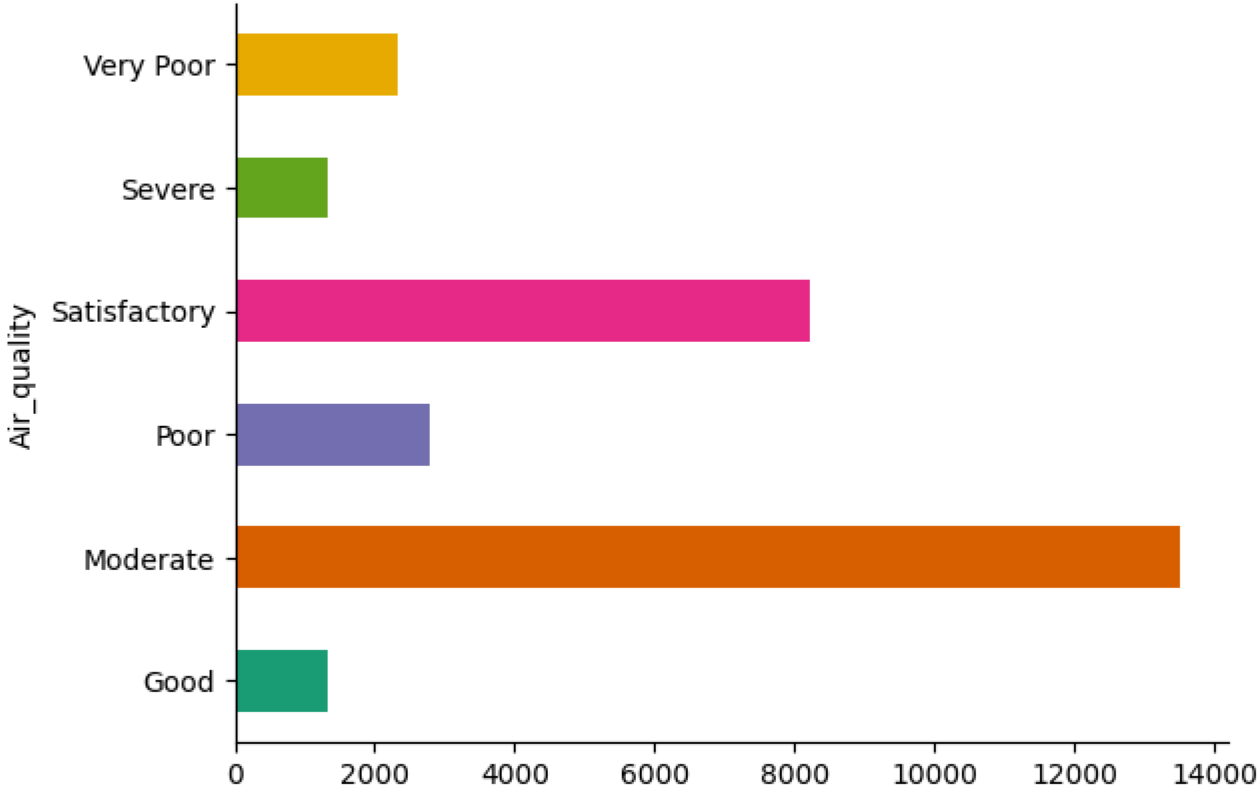
```
print(f"The available data is between {data['Date'].min()} and {data['Date'].max()}")
```

```
The available data is between 2015-01-01 00:00:00 and 2020-07-01 00:00:00
```

```
data['Air_quality'].unique()
```

```
array(['Moderate', 'Poor', 'Very Poor', 'Severe', 'Satisfactory', 'Good'],
      dtype=object)
```

```
from matplotlib import pyplot as plt
import seaborn as sns
data.groupby('Air_quality').size().plot(kind='barh', color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right',]].set_visible(False)
```



```
data['Air_quality'].value_counts()
```

```
                count
Air_quality
Moderate        13510
Satisfactory     8224
Poor             2781
Very Poor        2337
Good             1341
Severe           1338

dtype: int64
```

```
data['City'].unique()
```

```
array(['Ahmedabad', 'Aizawl', 'Amaravati', 'Amritsar', 'Bengaluru',
       'Bhopal', 'Brajrajnagar', 'Chandigarh', 'Chennai', 'Coimbatore',
       'Delhi', 'Ernakulam', 'Gurugram', 'Guwahati', 'Hyderabad',
       'Jaipur', 'Jorapokhar', 'Kochi', 'Kolkata', 'Lucknow', 'Mumbai',
       'Patna', 'Shillong', 'Talcher', 'Thiruvananthapuram',
       'Visakhapatnam'], dtype=object)
```

```
data['City'].value_counts()
```

	count
City	
Ahmedabad	2009
Delhi	2009
Mumbai	2009
Bengaluru	2009
Lucknow	2009
Chennai	2009
Hyderabad	2006
Patna	1858
Gurugram	1679
Visakhapatnam	1462
Amritsar	1221
Jorapokhar	1169
Jaipur	1114
Thiruvananthapuram	1112
Amaravati	951
Brajrajnagar	938
Talcher	925
Kolkata	814
Guwahati	502
Coimbatore	386
Shillong	310

Chandigarh	304
Bhopal	289
Ernakulam	162
Kochi	162
Aizawl	113

dtype: int64

```
print(data.isnull().sum())
```

```
City          0
Date          0
PM2.5         0
PM10          0
NO            0
NO2           0
NOx           0
NH3           0
CO            0
SO2           0
O3            0
Benzene       0
Toluene       0
Xylene        0
AQI           0
Air_quality   0
dtype: int64
```

```
data['Air_quality'] = data['Air_quality'].map({'Severe':0,'Very Poor': 1,'Poor':2, 'Moderate':3,'Satisfactory':4,'Good':5})
```

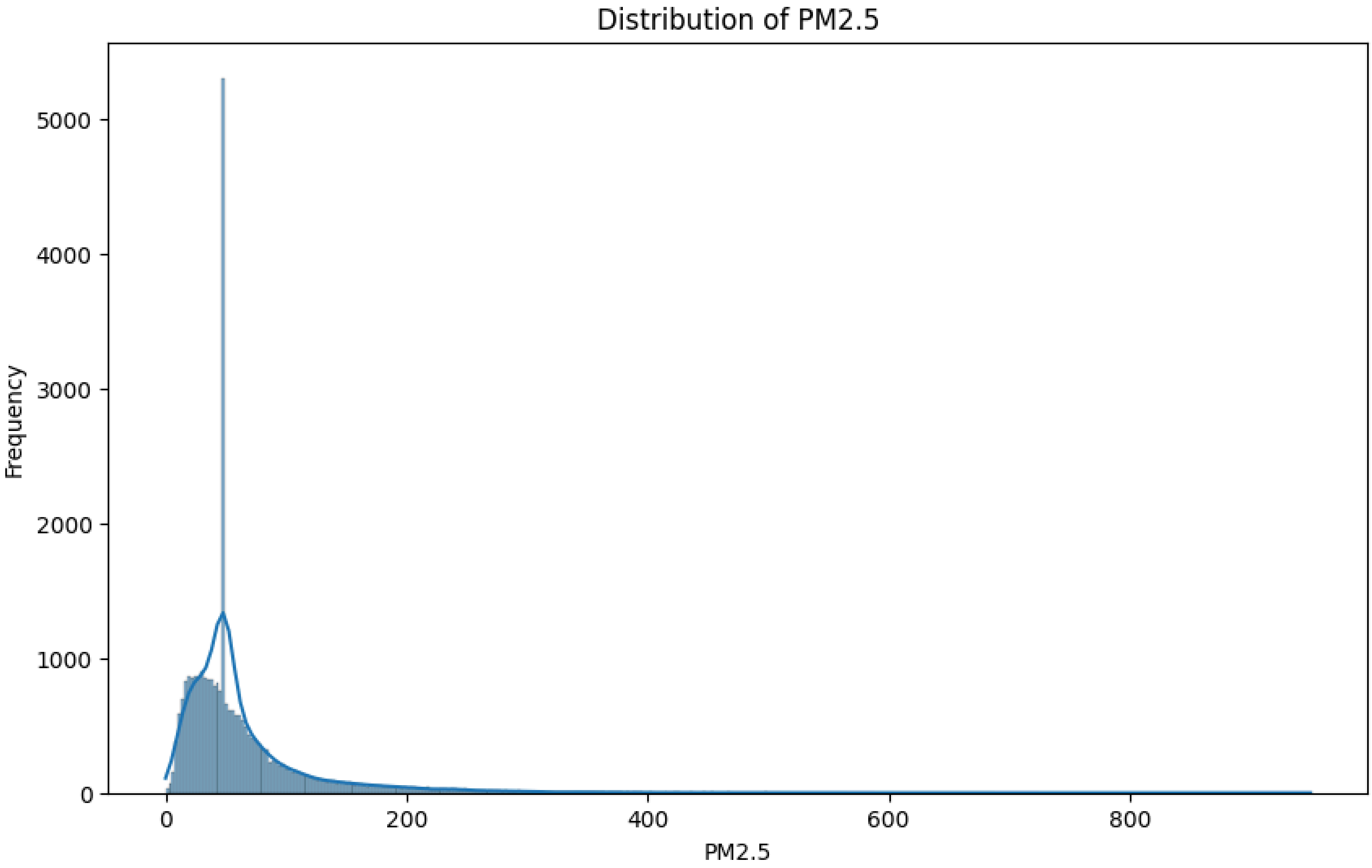
```
# Convert Date column to pandas datetime format
#data['Date'] = pd.to_datetime(data['Date'])

# Extract additional time-related features
#data['year'] = data['Date'].dt.year
```

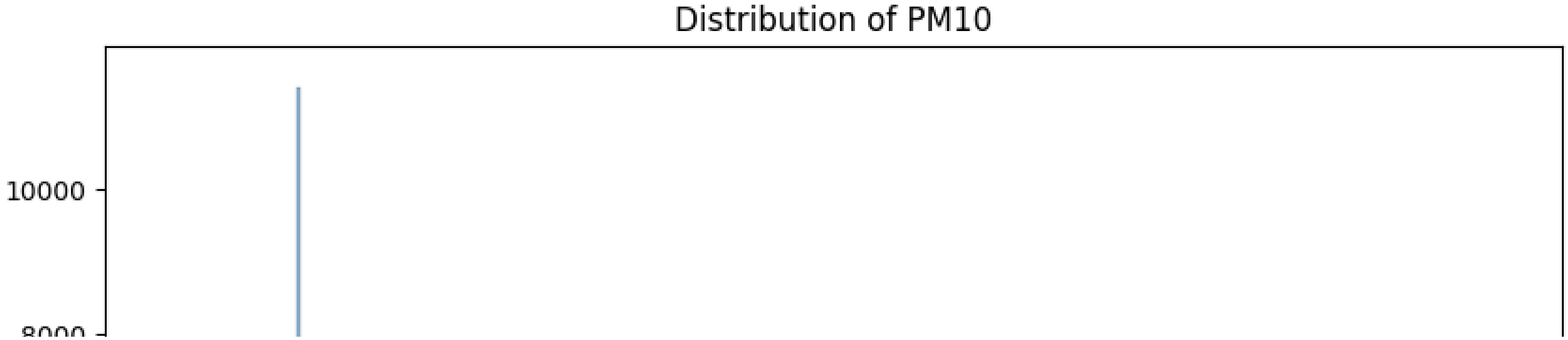
```
#data['month'] = data['Date'].dt.month
#data['day'] = data['Date'].dt.day

# Display updated dataset
#data.head()
```

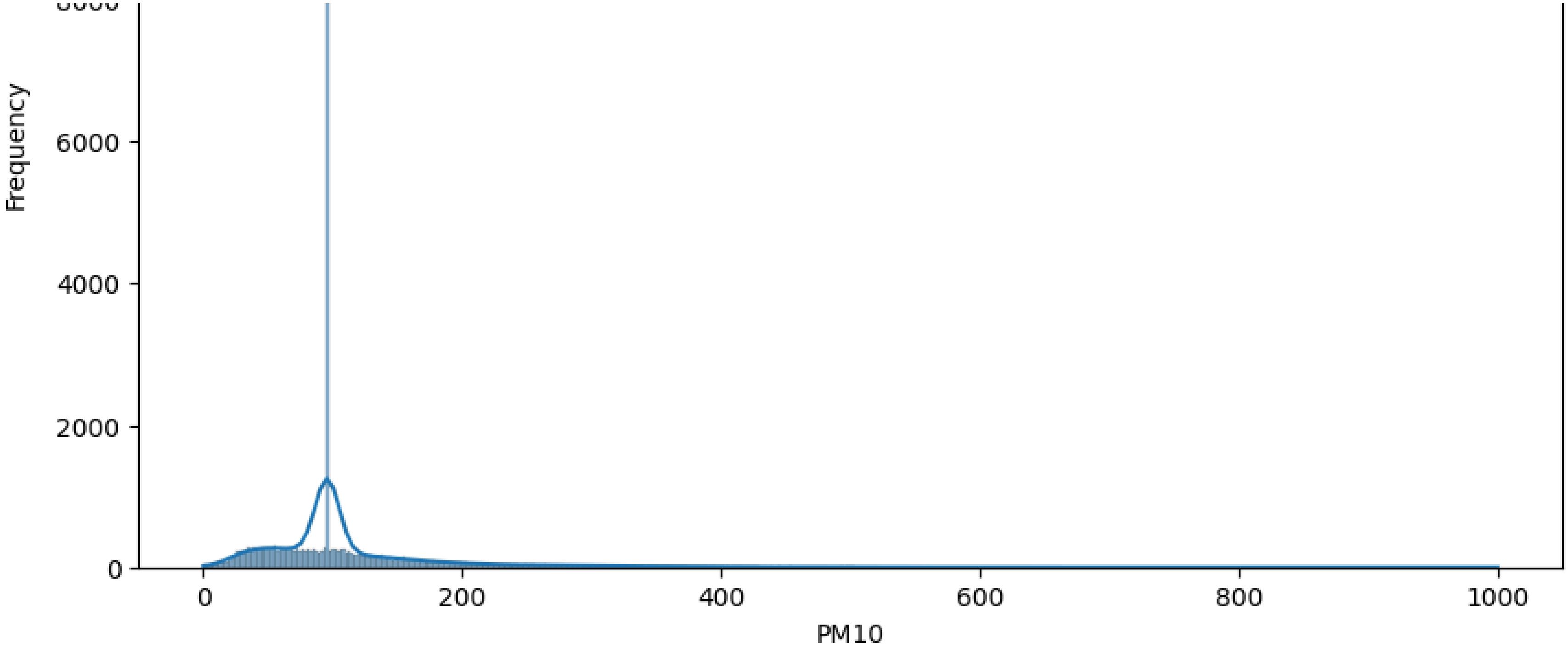
```
data.describe()
```

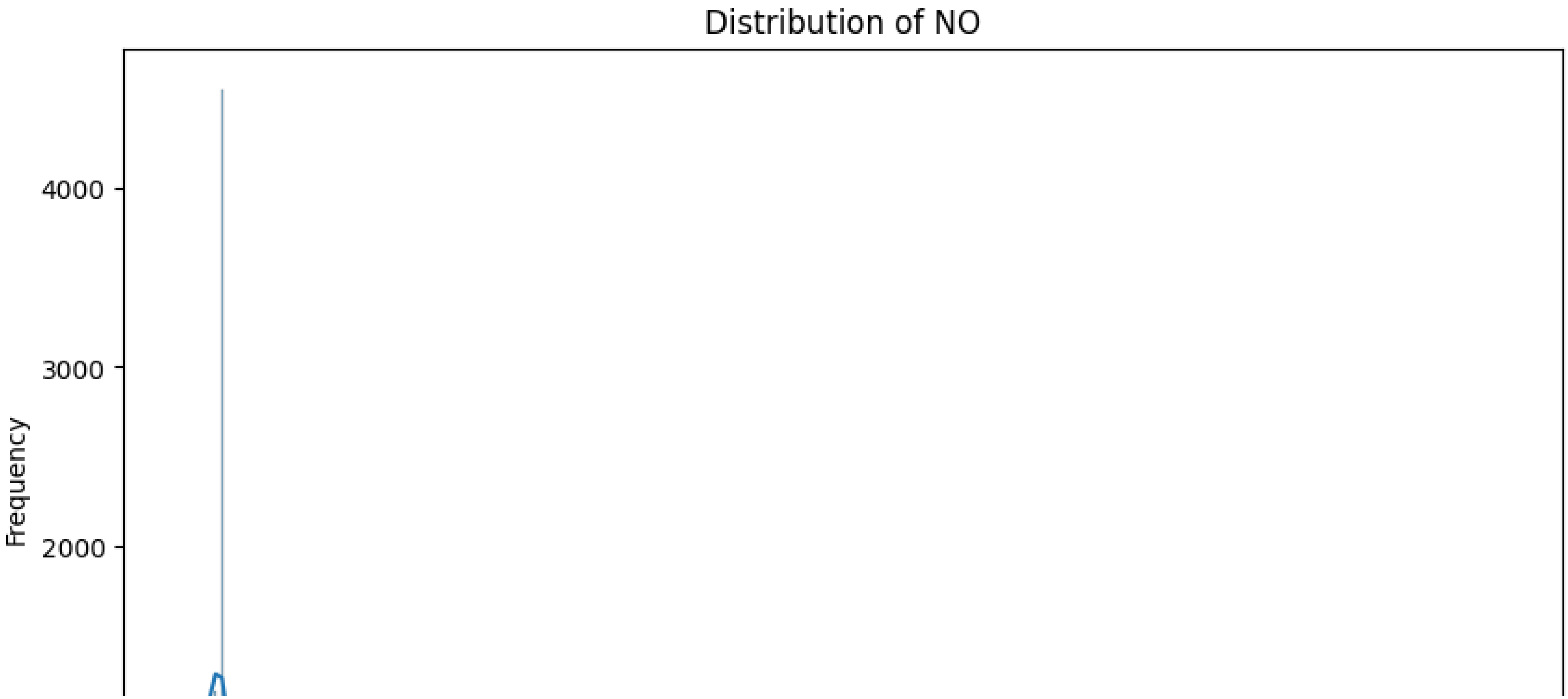
	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO
count	29531	29531.000000	29531.000000	29531.000000	29531.000000	29531.000000	29531.000000	29531.000000
mean	2018-05-14 05:40:15.807118080	64.510857	109.659366	16.642601	27.726576	31.063568	20.813789	2.153872
min	2015-01-01 00:00:00	0.040000	0.010000	0.020000	0.010000	0.000000	0.010000	0.000000
25%	2017-04-16 00:00:00	32.150000	79.315000	6.210000	12.980000	14.670000	12.040000	0.540000
50%	2018-08-05 00:00:00	48.570000	95.680000	9.890000	21.690000	23.520000	15.850000	0.890000
75%	2019-09-03 00:00:00	72.450000	111.880000	17.570000	34.665000	36.015000	21.755000	1.380000
max	2020-07-01 00:00:00	949.990000	1000.000000	390.680000	362.210000	467.630000	352.890000	175.810000
std	NaN	59.807551	72.324020	21.506064	23.050531	29.477748	21.028862	6.724660

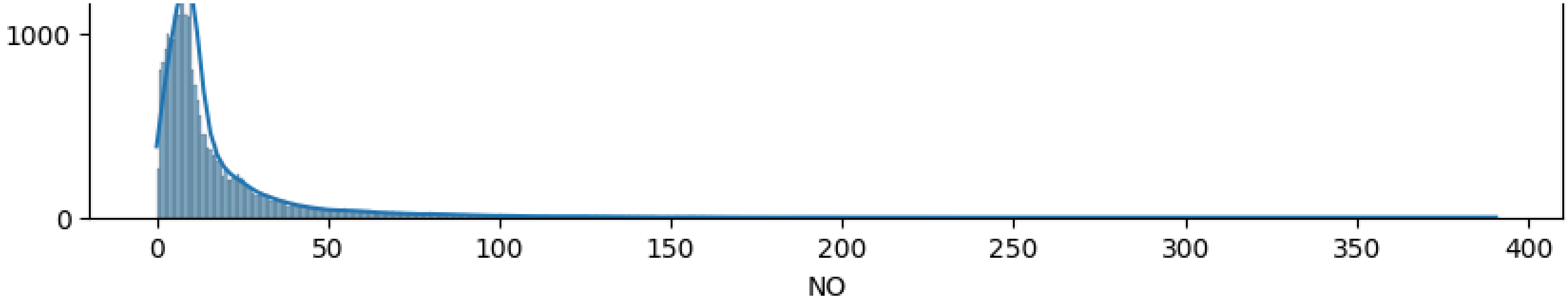
```
# Distribution of air pollutants
pollutants = ['PM2.5','PM10','NO', 'NO2','NOx','NH3','CO','SO2','O3','Benzene','Toluene','Xylene','AQI','Air_quality']
for pollutant in pollutants:
    plt.figure(figsize=(10, 6))
    sns.histplot(data[pollutant], kde=True)
    plt.title(f'Distribution of {pollutant}')
    plt.xlabel(pollutant)
    plt.ylabel('Frequency')
    plt.show()
```

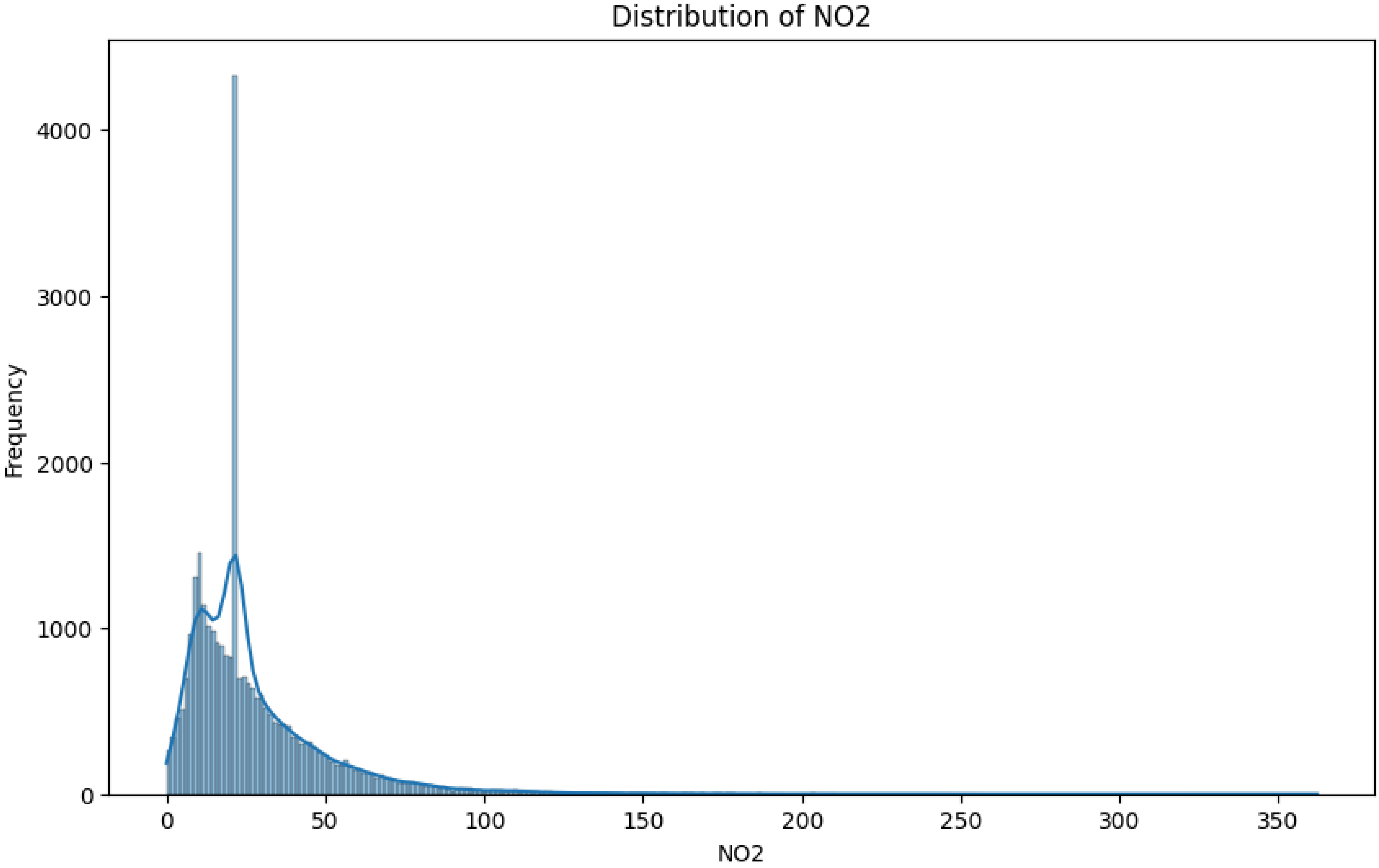

Distribution of PM2.5
Frequency
0
1000
2000
3000
4000
5000
0
200
400
600
800
PM2.5


Distribution of PM10
10000
8000

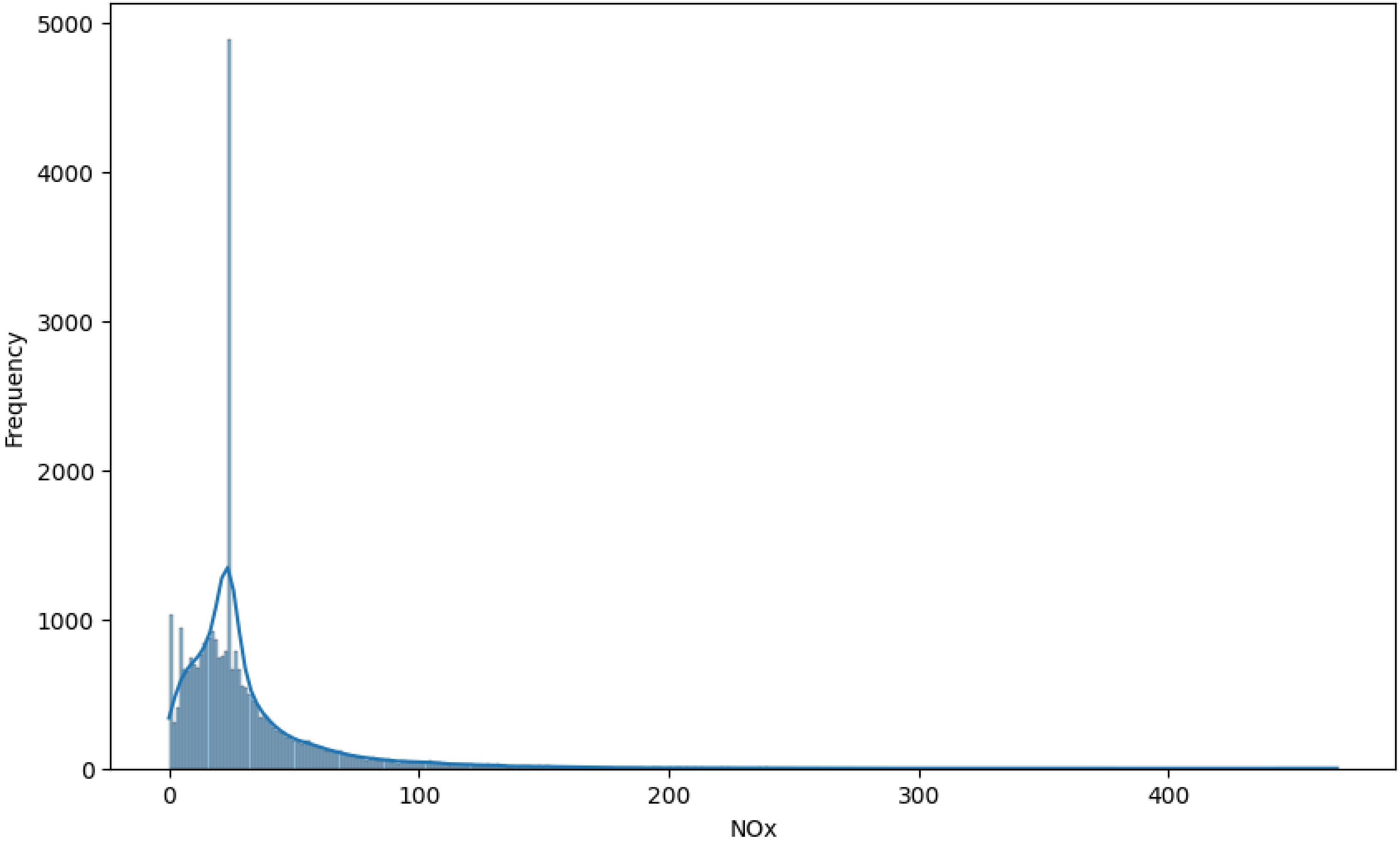

8000
6000
4000
2000
0
Frequency
0
200
400
600
800
1000
PM10

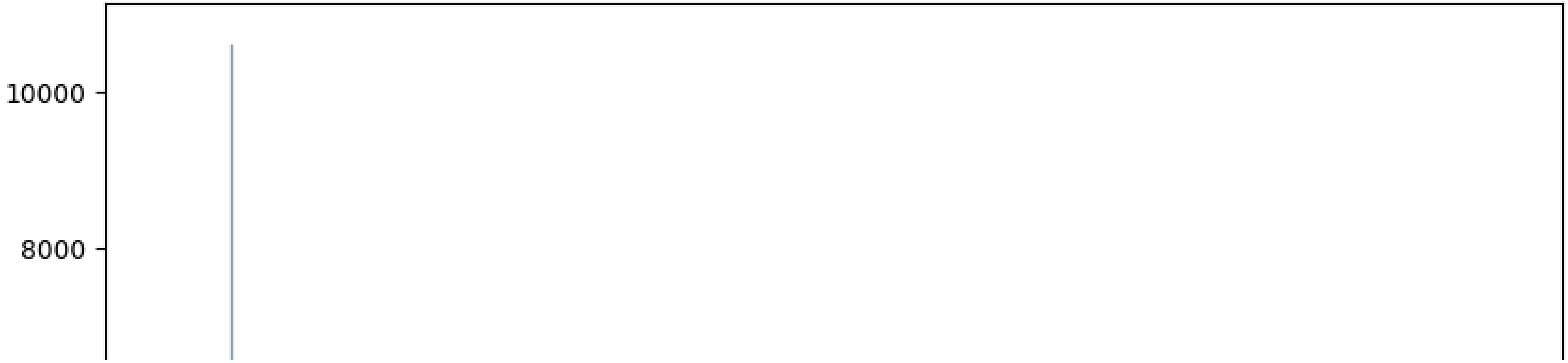

Distribution of NO
4000
3000
2000
Frequency

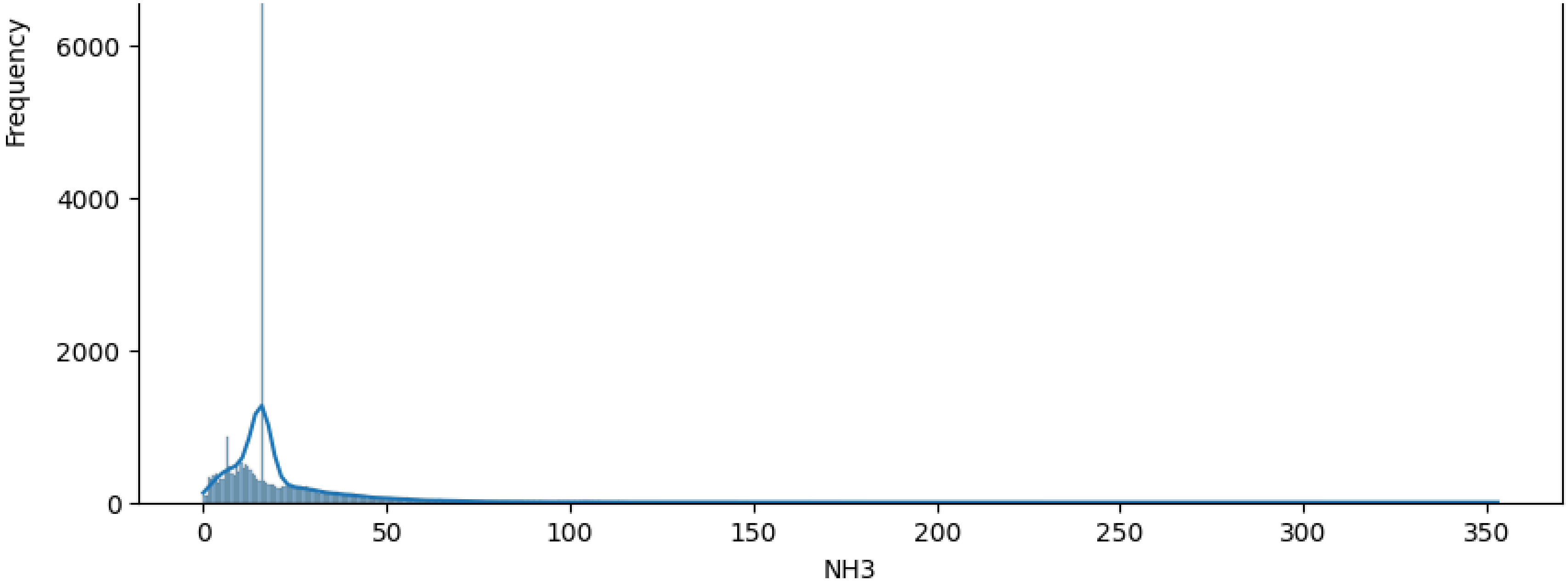


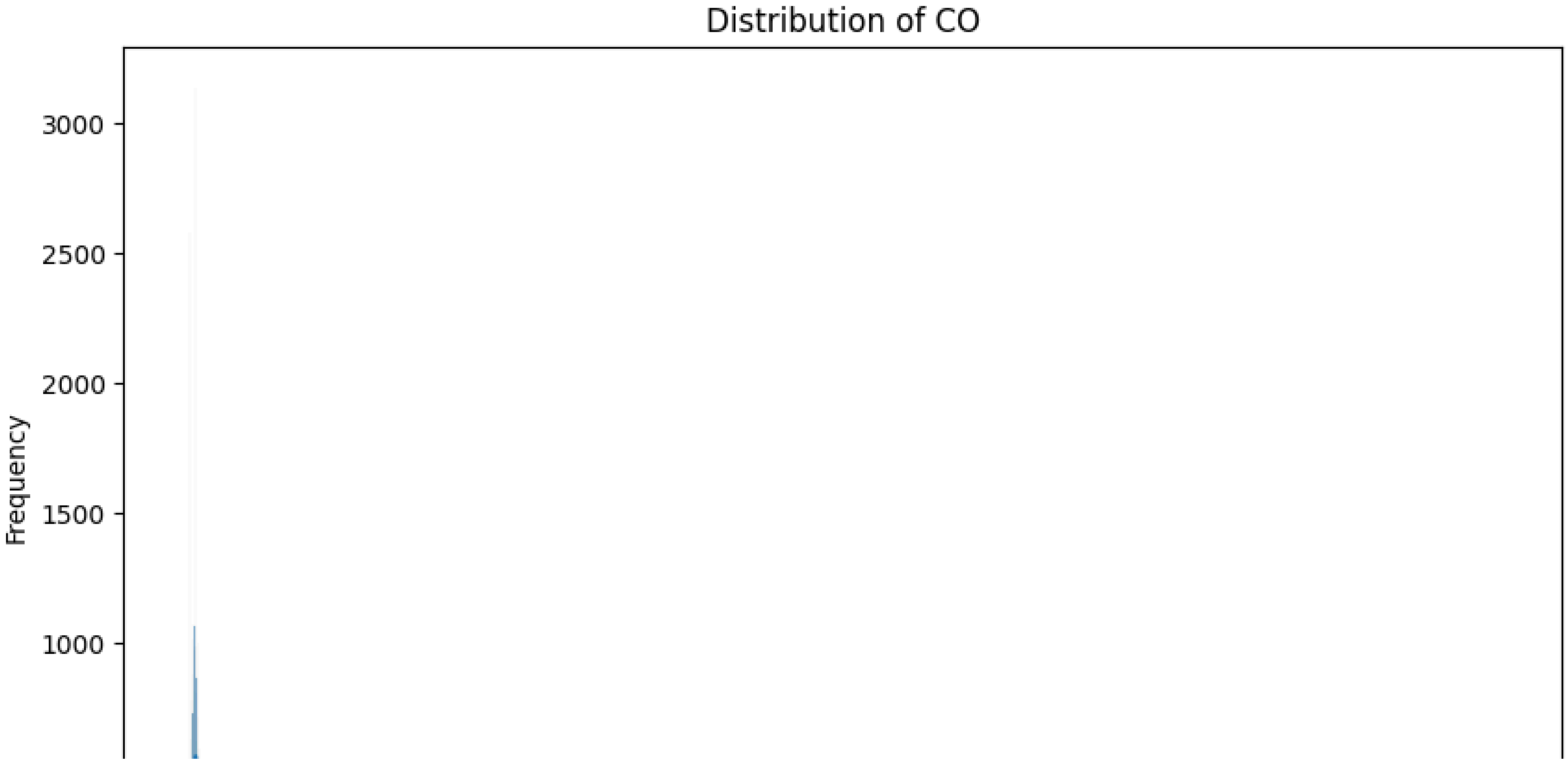


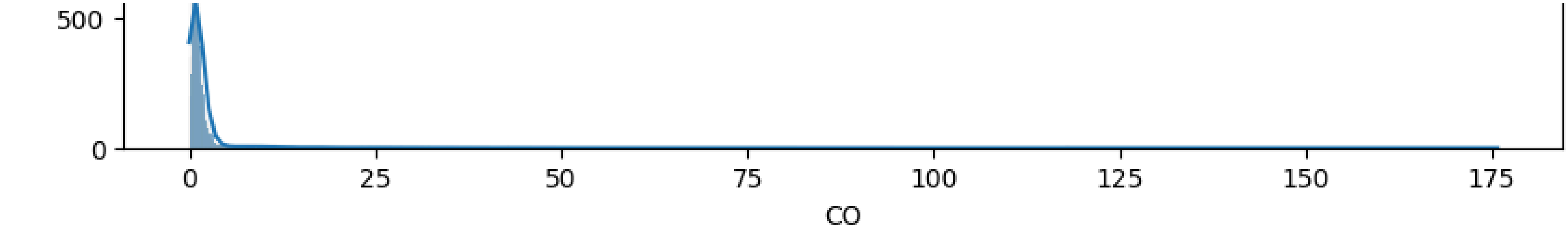
Distribution of NOx



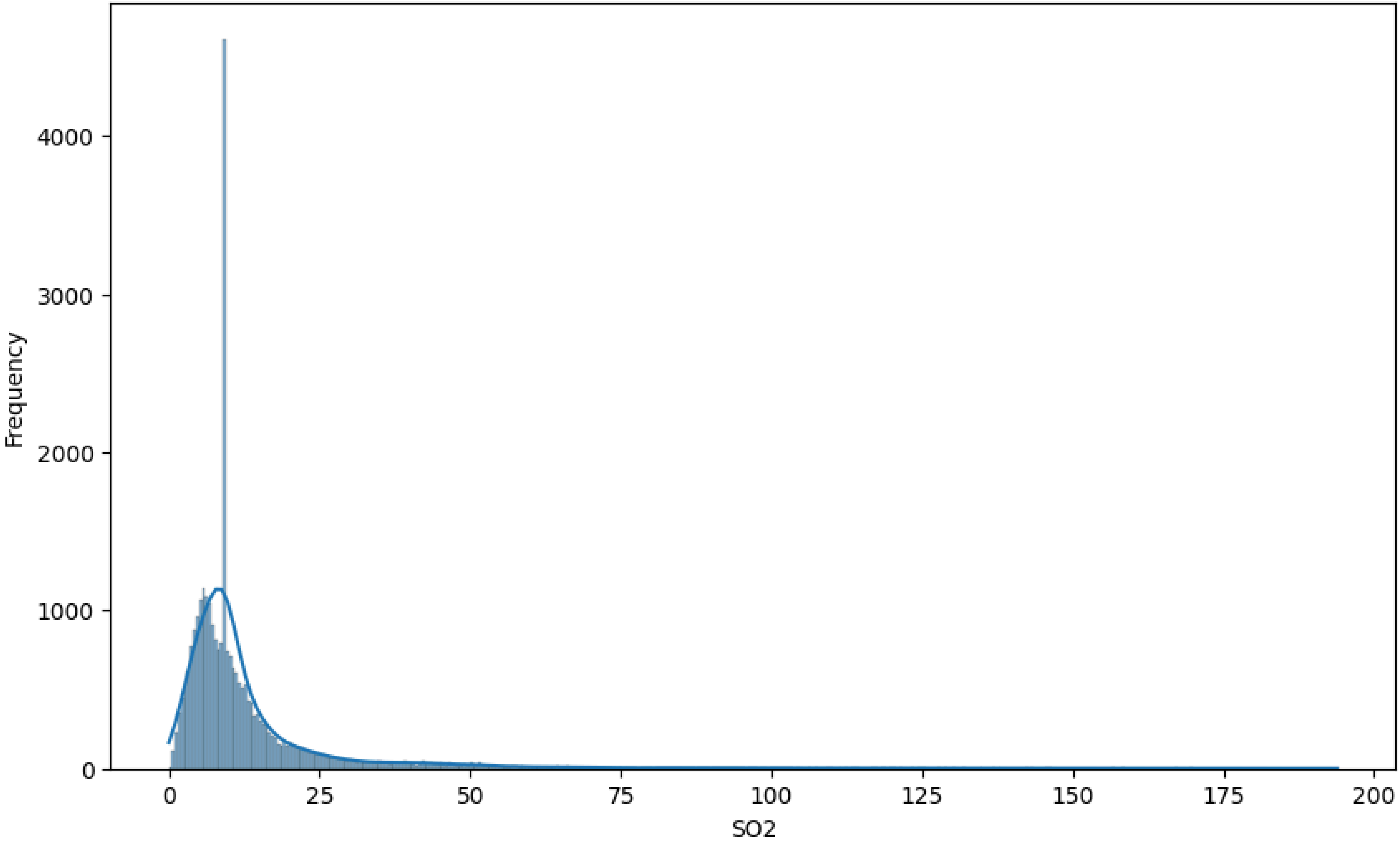

Distribution of NH3


Frequency
6000
4000
2000
0
0
50
100
150
200
250
300
350
NH3

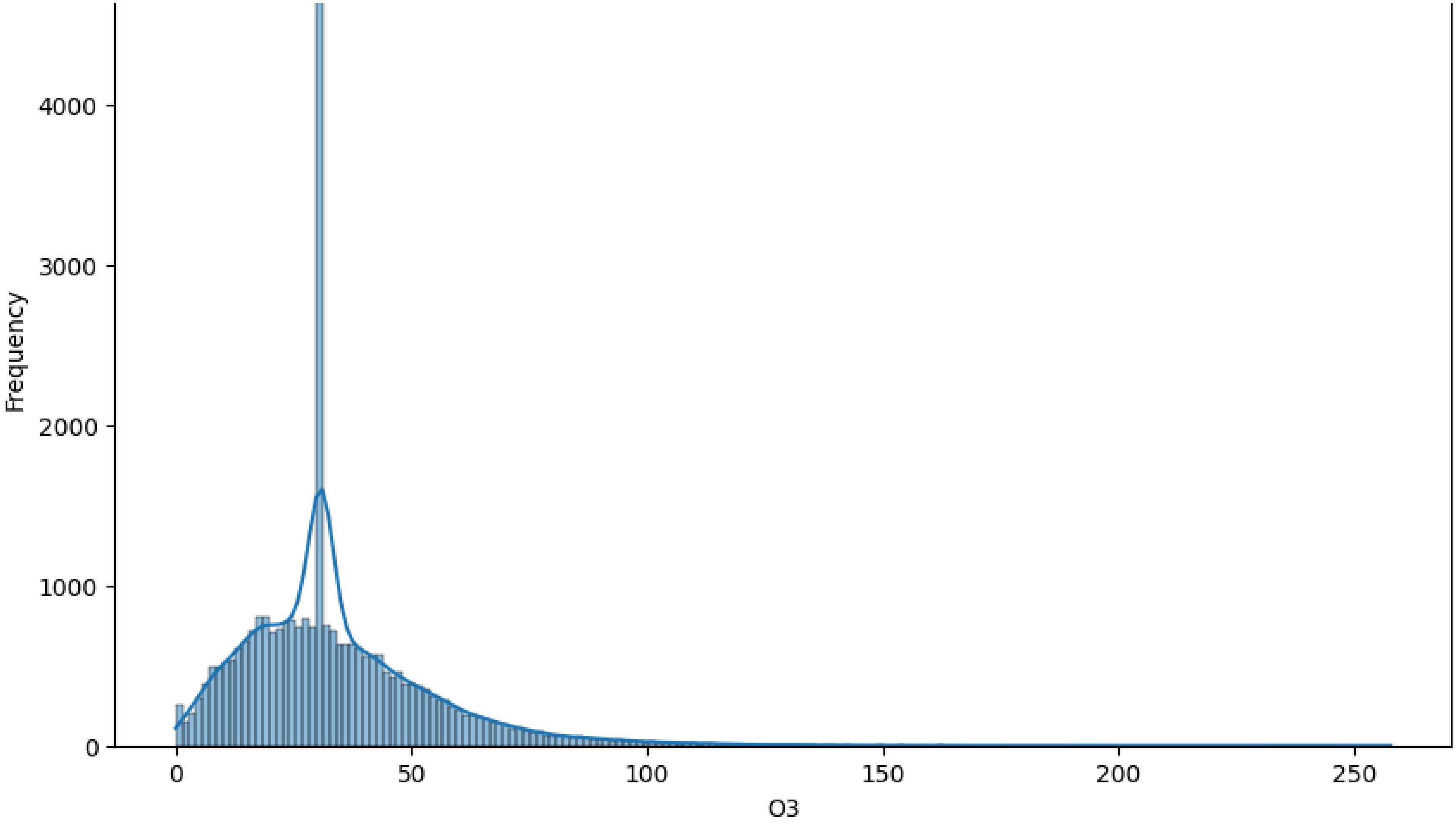

Distribution of CO
Frequency
3000
2500
2000
1500
1000

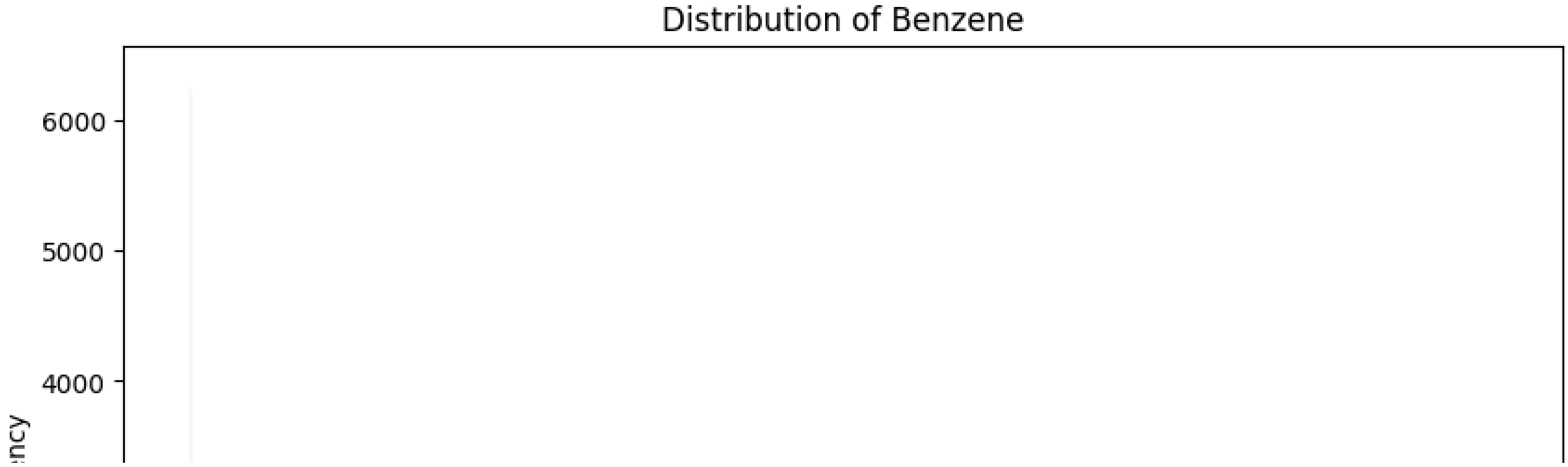


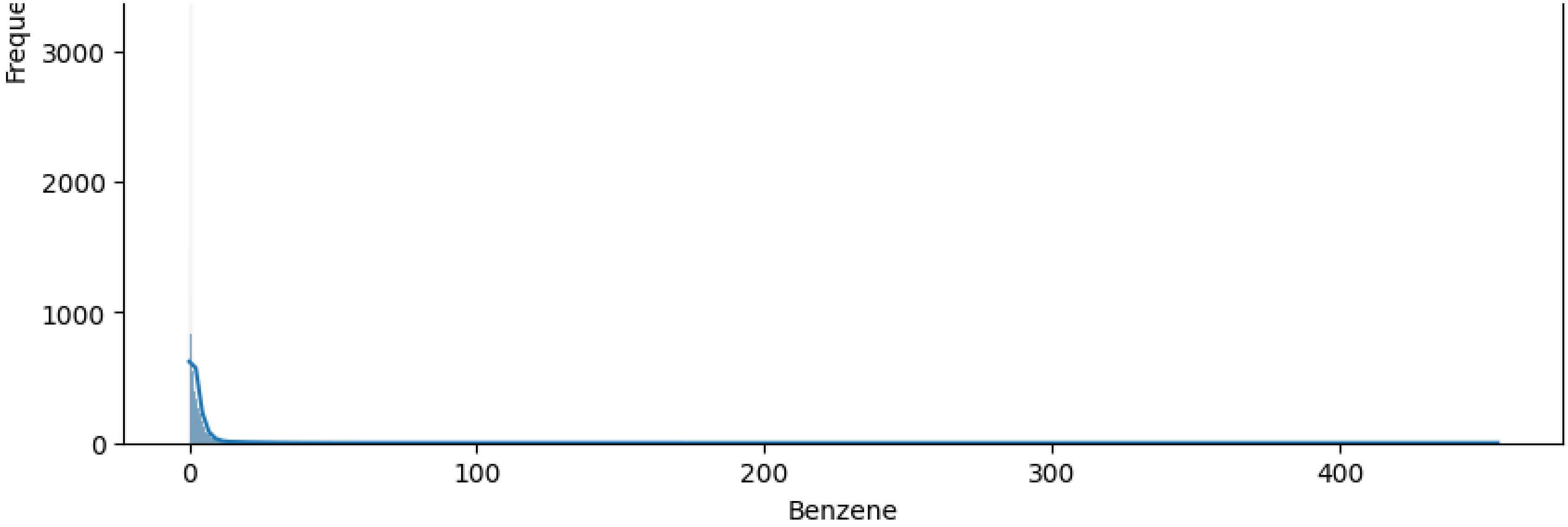
Distribution of SO2

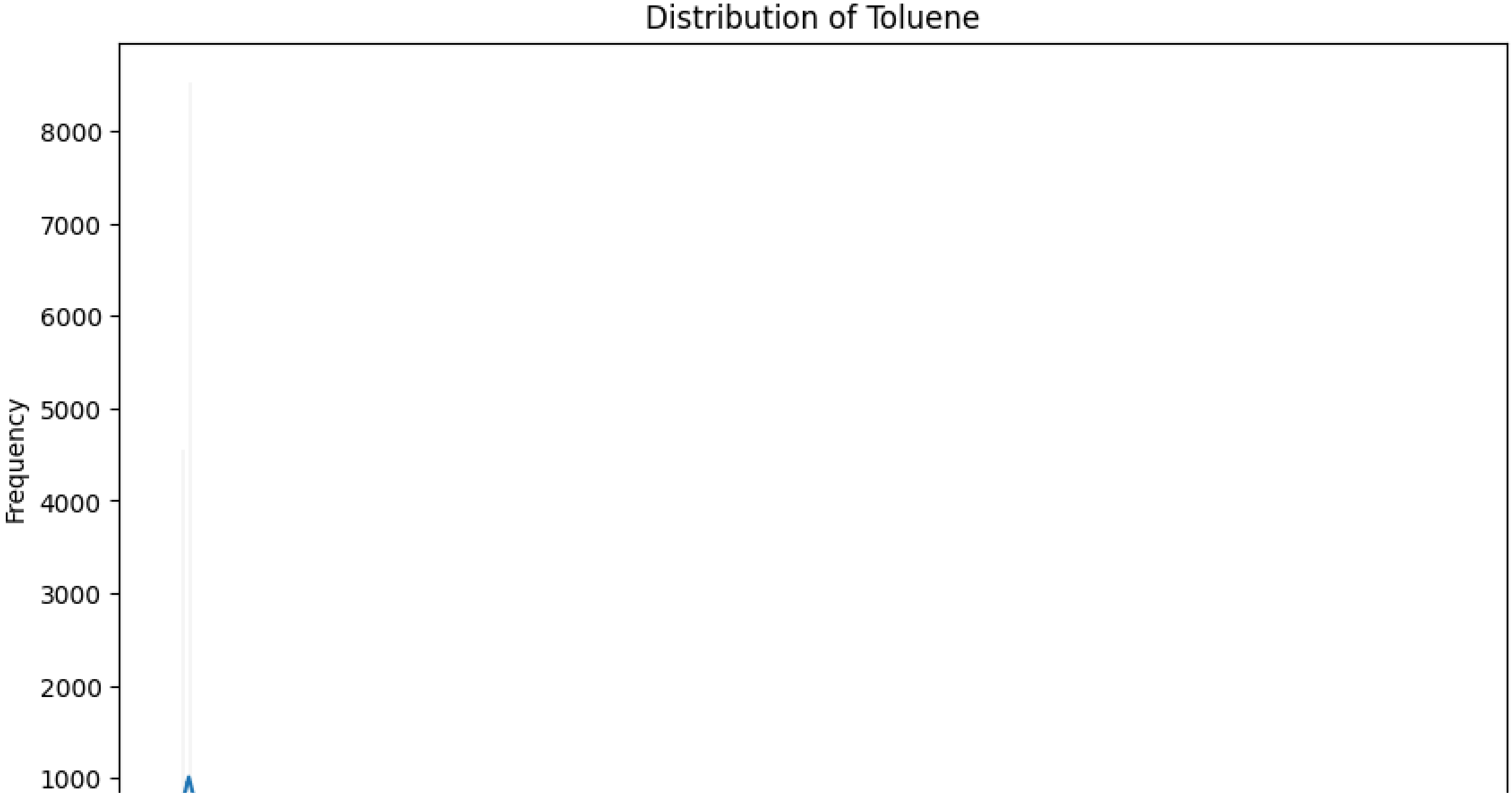


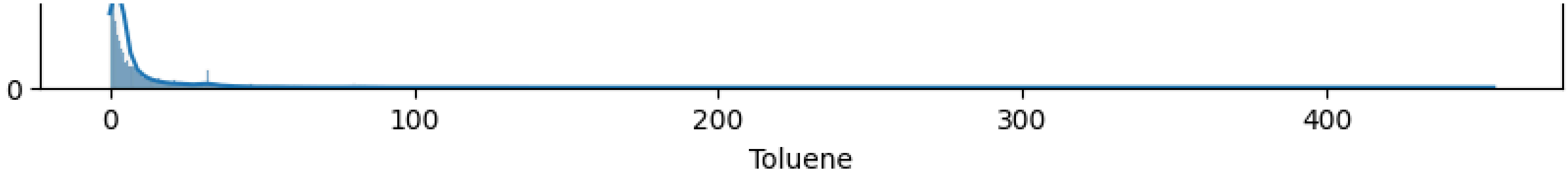
Distribution of O3

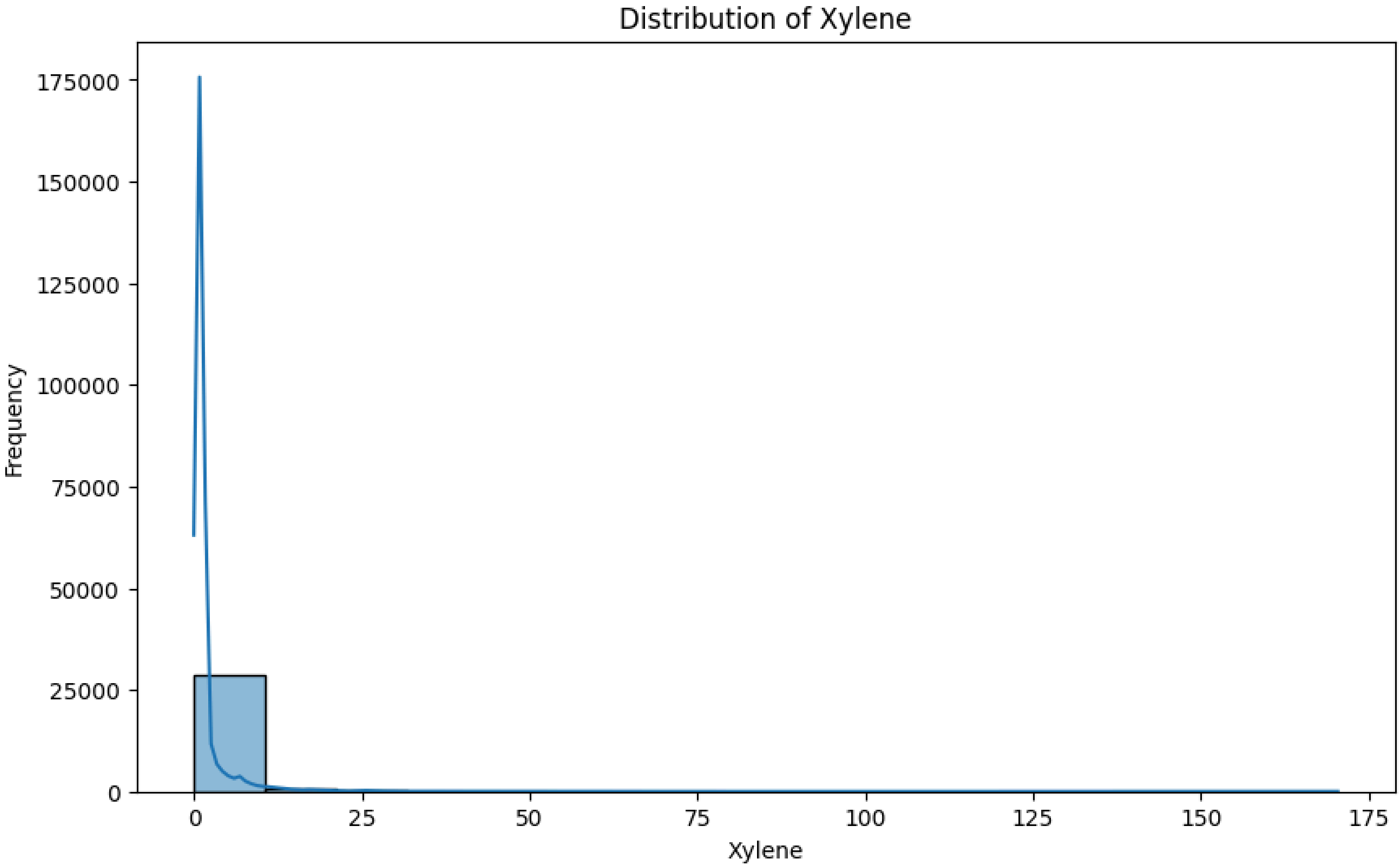

Frequency
4000
3000
2000
1000
0
0
50
100
150
200
250
O3

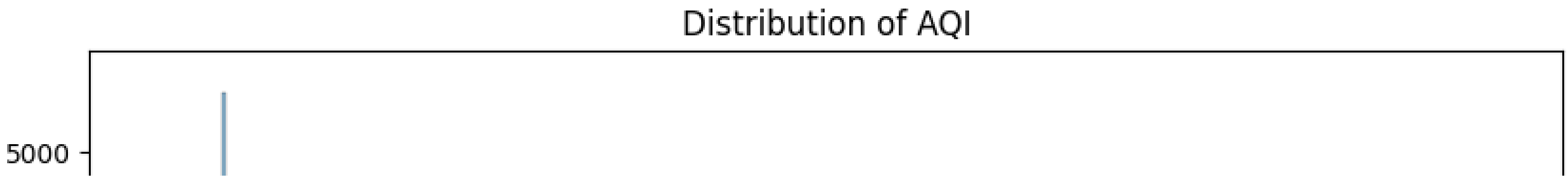

Distribution of Benzene
6000
5000
4000
ncy

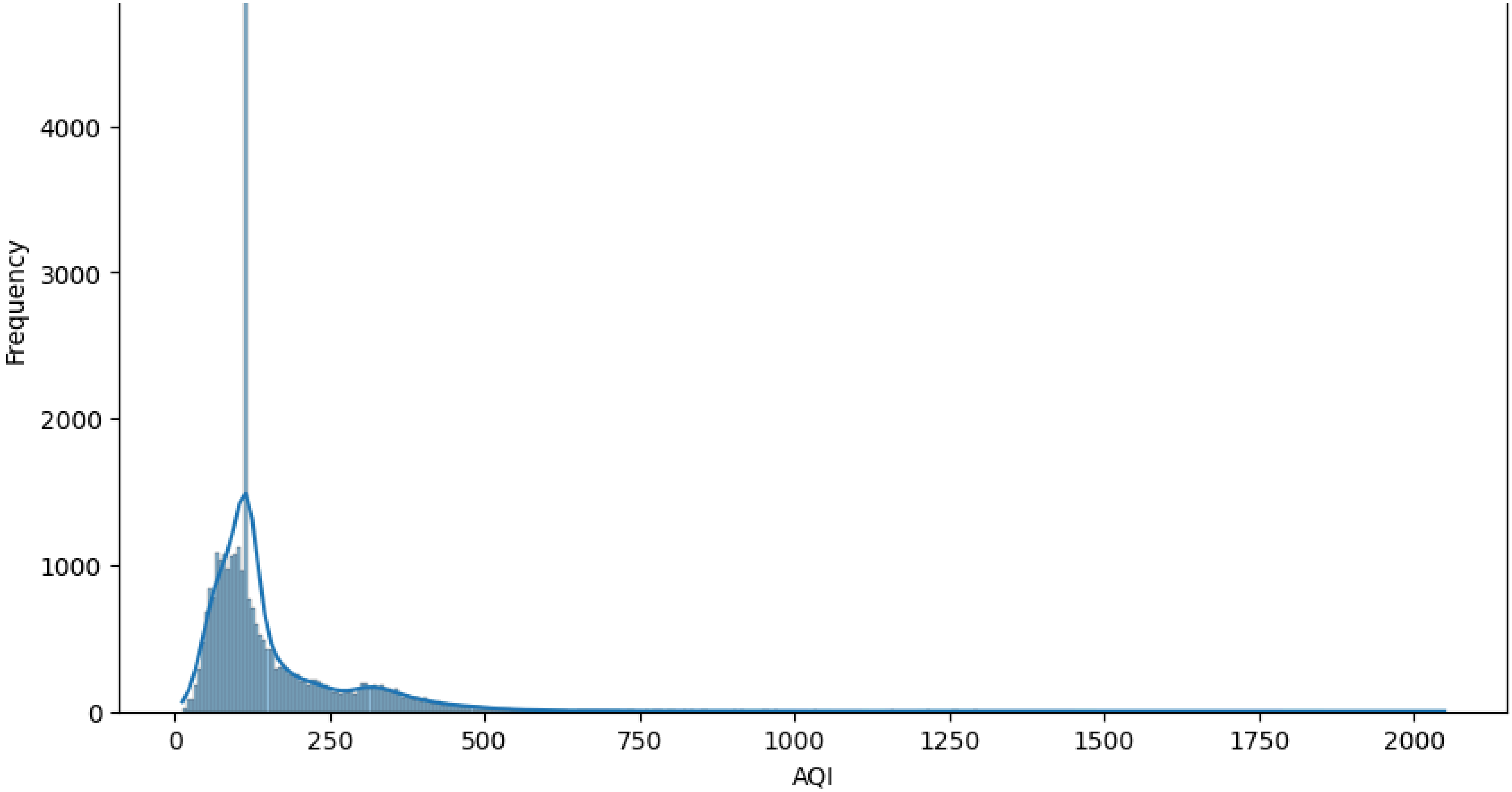

Freque
3000
2000
1000
0
0
100
200
300
400
Benzene

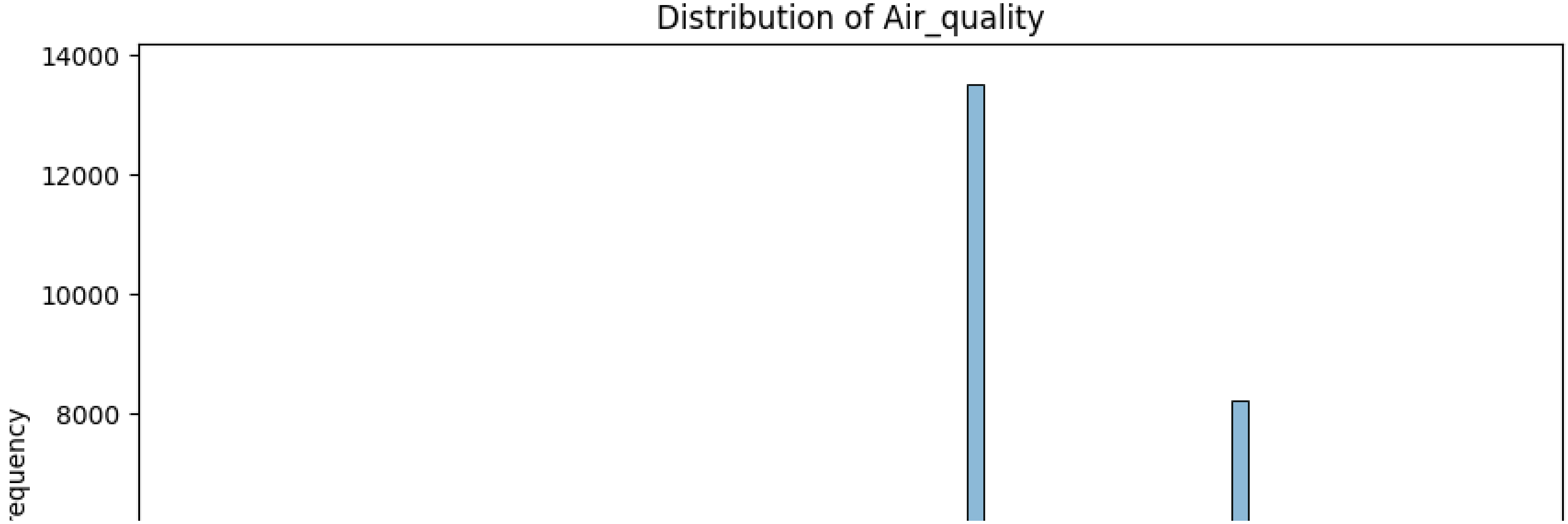

Distribution of Toluene
Frequency
8000
7000
6000
5000
4000
3000
2000
1000

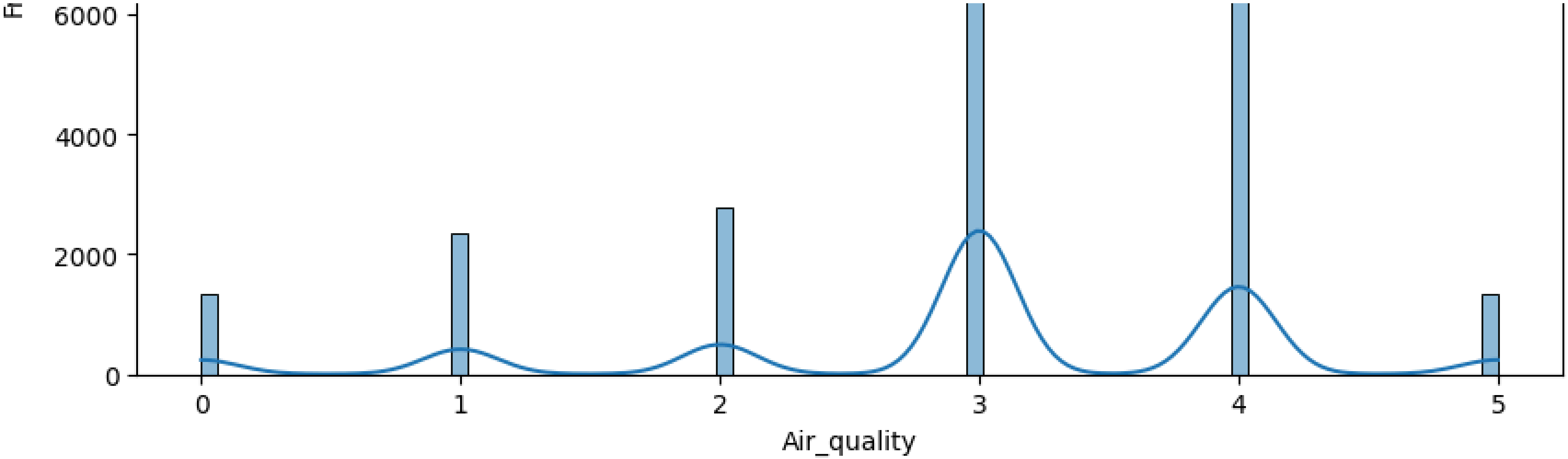



Distribution of Xylene

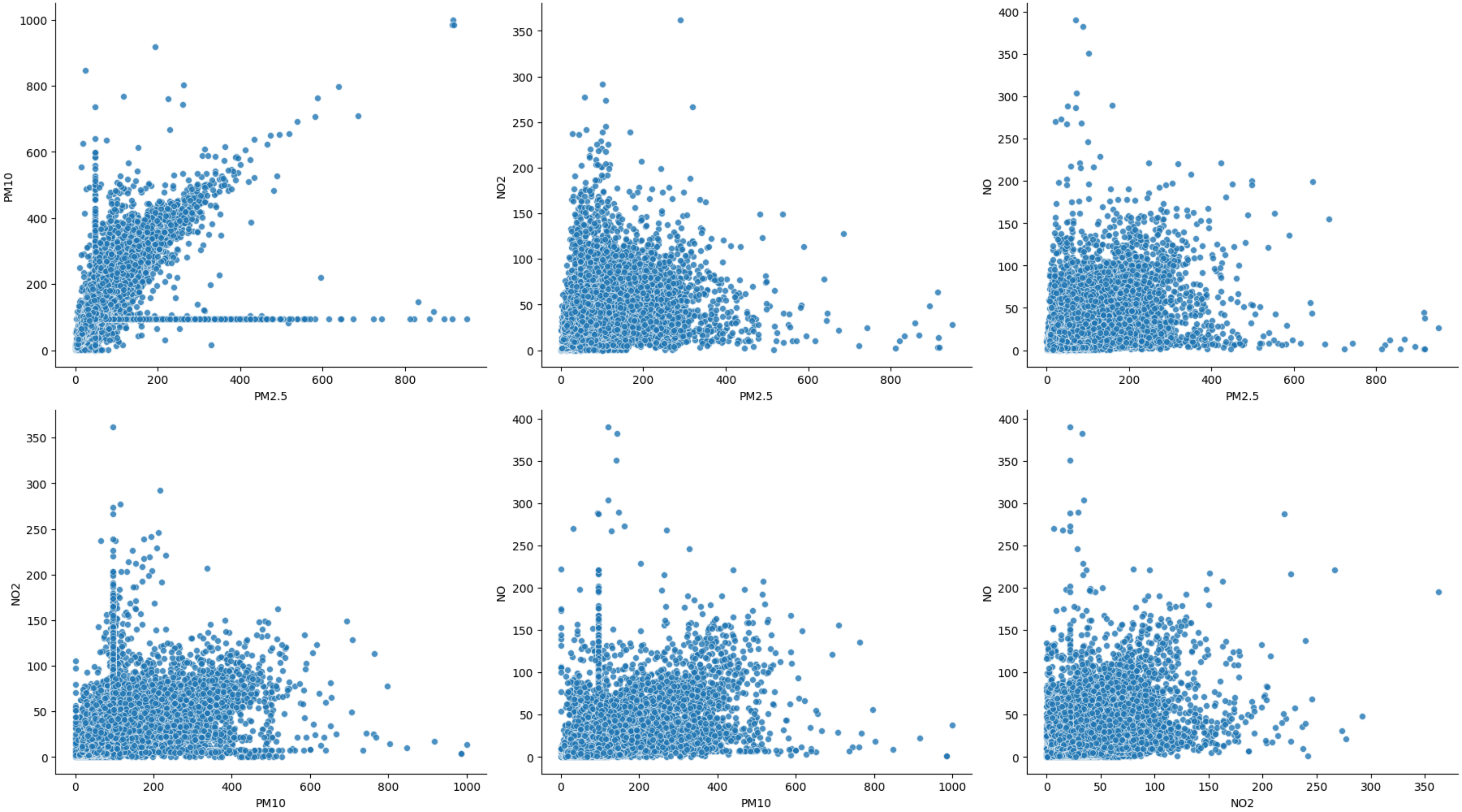

Distribution of AQI


Frequency
4000
3000
2000
1000
0
0
250
500
750
1000
1250
1500
1750
2000
AQI


Distribution of Air_quality
14000
12000
10000
8000
requency

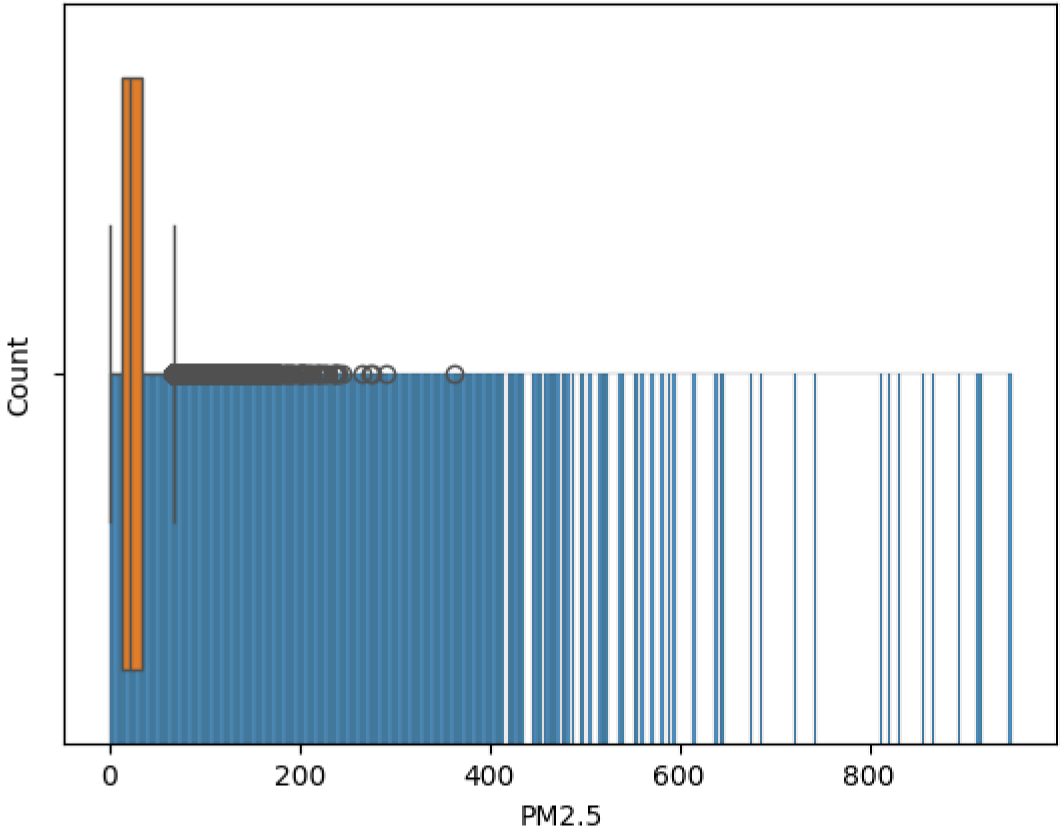

6000
4000
2000
0
0
1
2
3
4
5
Air_quality

```
# Define the pairs of pollutants to plot
pairs = [
    ('PM2.5', 'PM10'),
    ('PM2.5', 'NO2'),
    ('PM2.5', 'NO'),
    ('PM10', 'NO2'),
    ('PM10', 'NO'),
    ('NO2', 'NO')
]

# Create a 2x3 grid of subplots
fig, axes = plt.subplots(2, 3, figsize=(18, 10), constrained_layout=True)

# Flatten the 2x3 grid of axes
axes = axes.flatten()

# Loop through each pair and corresponding subplot
for ax, (x, y) in zip(axes, pairs):
    sns.scatterplot(data=data, x=x, y=y, s=32, alpha=0.8, ax=ax)
    sns.despine(ax=ax)  # Remove top and right spines
    ax.set_xlabel(x)
    ax.set_ylabel(y)

# Show the plots
plt.show()
```


PM10
1000
800
600
400
200
0
0
200
400
600
800
PM2.5
NO2
350
300
250
200
150
100
50
0
0
200
400
600
800
PM2.5
NO
400
350
300
250
200
150
100
50
0
0
200
400
600
800
PM2.5
NO2
350
300
250
200
150
100
50
0
0
200
400
600
800
1000
PM10
NO
400
350
300
250
200
150
100
50
0
0
200
400
600
800
1000
PM10
NO
400
350
300
250
200
150
100
50
0
0
50
100
150
200
250
300
350
NO2

```
# Distribution plots
print(sns.histplot(data['PM2.5']))
print(sns.boxplot(x=data['NO2']))
```

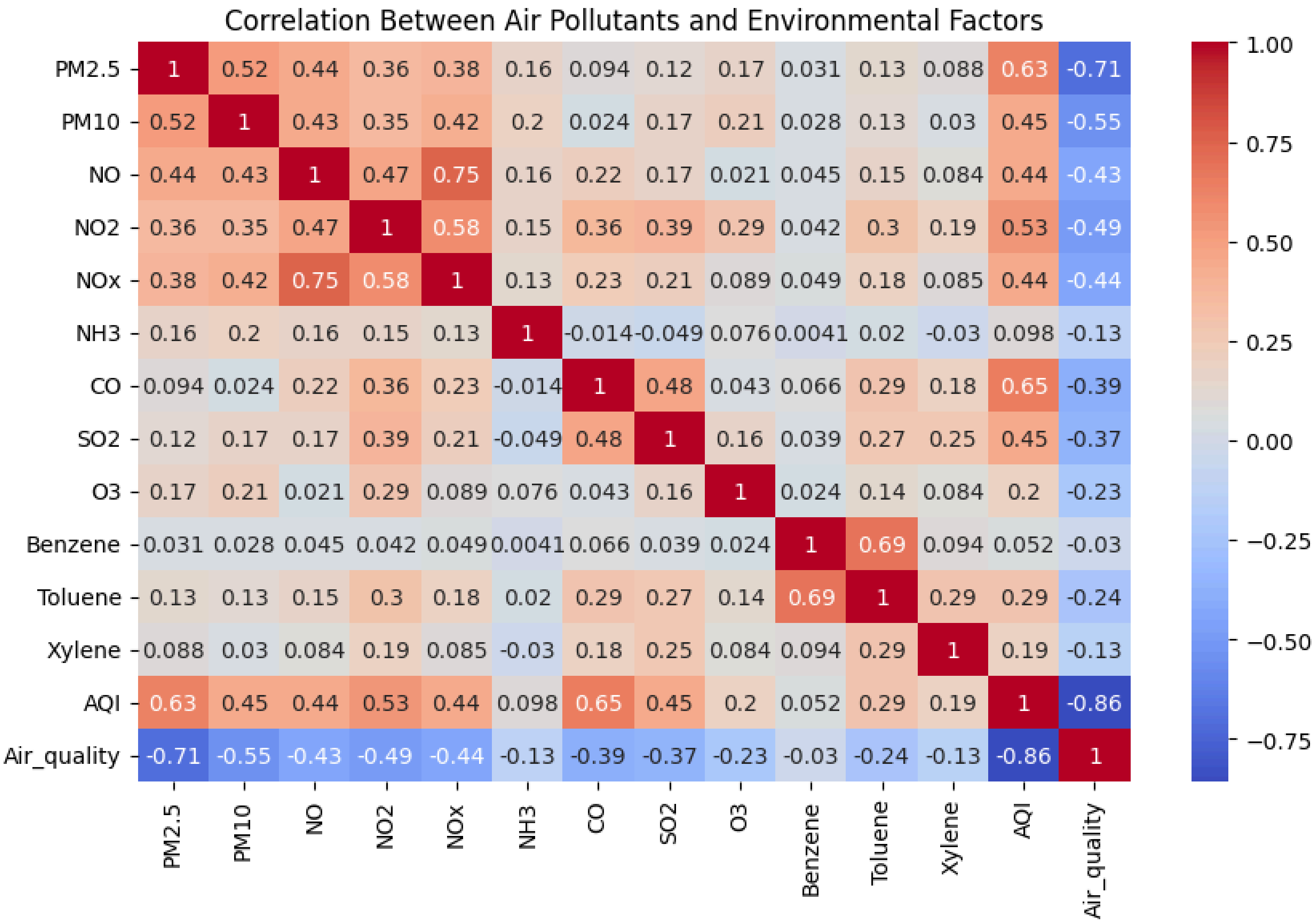
```
Axes(0.125,0.11;0.775x0.77)
Axes(0.125,0.11;0.775x0.77)
```



```
data1 = data.drop(columns=['City','Date'])
```

```
# Create a correlation matrix
corr_matrix = data1.corr()
# Plot the heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
```

```
plt.title('Correlation Between Air Pollutants and Environmental Factors')
plt.show()
```



```
from matplotlib import pyplot as plt
import seaborn as sns
```

```
def _plot_series(series, series_name, series_index, pollutant, ax):
    palette = list(sns.palettes.mpl_palette('Dark2'))
    xs = series['Date']
    ys = series[pollutant]
    ax.plot(xs, ys, label=series_name, color=palette[series_index % len(palette)]

def plot_multiple_pollutants(or_data):
    pollutants = ['PM2.5', 'PM10', 'NO2']
    fig, axes = plt.subplots(3, 1, figsize=(10, 15.6), constrained_layout=True)

    df_sorted = data.sort_values('Date', ascending=True)
    for i, (series_name, series) in enumerate(df_sorted.groupby('AQI_Bucket')):
        for j, pollutant in enumerate(pollutants):
            _plot_series(series, series_name, i, pollutant, axes[j])

    for ax, pollutant in zip(axes, pollutants):
        ax.legend(title='Air_quality', bbox_to_anchor=(1, 1), loc='upper left')
        sns.despine(fig=fig, ax=ax)
        ax.set_xlabel('Date')
        ax.set_ylabel(pollutant)

    plt.show()
```

```
#df = data[['SO2','year','City']].groupby(["year"]).median().reset_index().sort_values(by='year',ascending=False)
#f,ax=plt.subplots(figsize=(15,5))
#sns.pointplot(x='year', y='SO2', data=df)
```

Model Building

```
X = data.drop(columns = ['City','AQI','Air_quality','Date'])
```